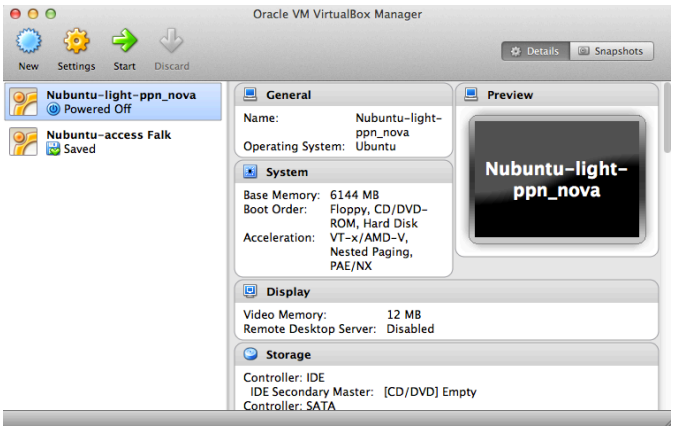
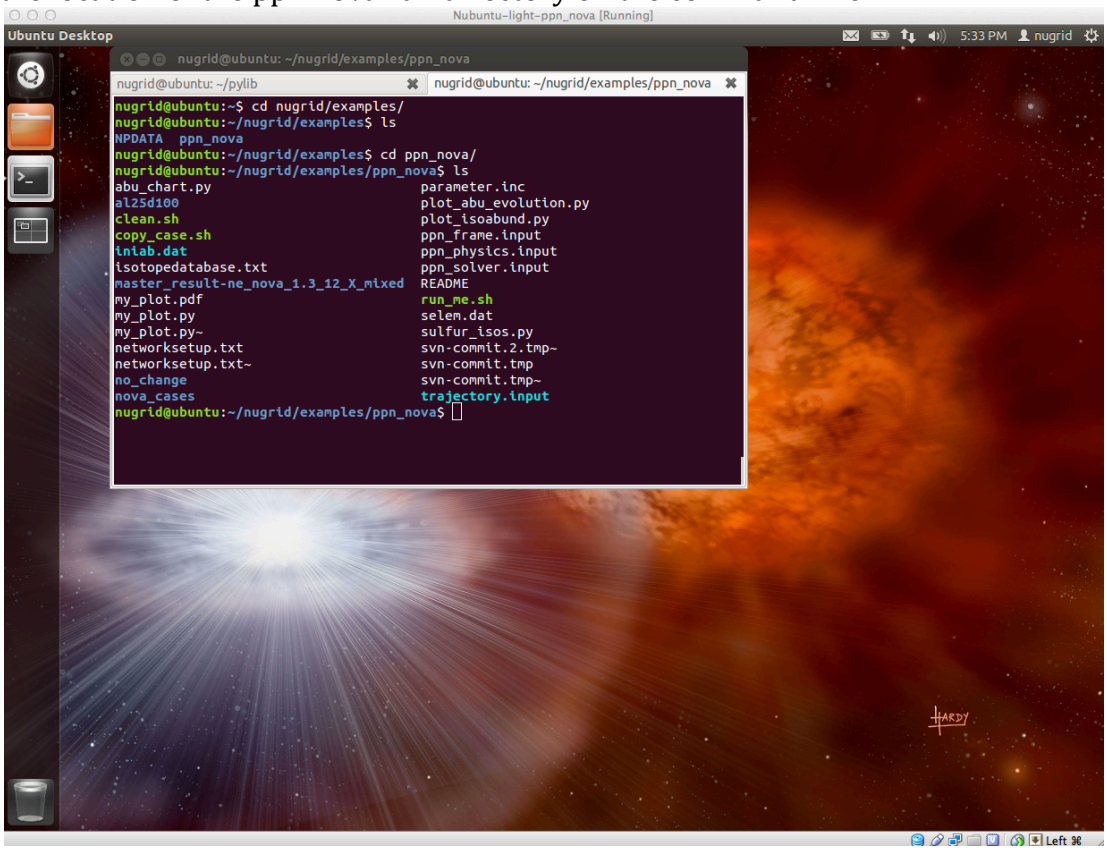
A step-by-step guide to perform a reaction rate sensitivity test with the NuGrid-JINA-TRIUMF nova ppn framework

The goal of this HowTo is to demonstrate how to perform a simple task using the NuGrid nova ppn framework in a VirtualBox machine. Specifically we want ton show how Fig.15 in Denissenkov etal. 2013 (arXiv:1303.6265) can be made. It is assumed that the VirtualBox software (https://www.virtualbox.org) is installed and the Nubuntu-light-ppn_nova machine has been downloaded and started. In order to open a virtual box machine for the first time double click on the vbox file. The VirtualBox Manager window would then look like this:



After starting the Nubuntu-light-ppn_nova machine, and logging in (user name and passwd are 'nugrid') we can open a terminal in the virtual machine and navigate to the location of the ppn nova run directory on the command line:



In that directory you find a README file with useful information. In order to perform a standard run open the run script run_me.sh and select one of the available nova trajectories (this may be updated in the version you are using):

```
nugrid@ubuntu: ~/nugrid/examples/ppn_nova
File Edit Options Buffers Tools Sh-Script Help

# select one of the nova trajectories from the nova_cases directory:
#
# co_nova_1.15_10_B_mixed co_nova_1.15_12_X_mixed
# ne_nova_1.3_12_X_mixed ne_nova_1.15_12_X_mixed ne_nova_1.3_7_B_mixed
#
# The first number in the file name is the WD mass (in Msun), the second
# is its initial central temperature (in MK), the letter X means the
# mass accretion rate 2e-10 Msun/yr, while B means 1e-11 Msun/yr.
case=ne_nova_1.3_12_X_mixed

#### most of the time no intervention required below this line #####
```

Then just execute the run_me.sh script. After a couple of minutes you will get some output indicating that the standard plots are automatically performed:

```
nugrid@ubuntu: ~/nugrid/examples/ppn_nova
Using the following conditions:
        Atomic mass_range: 1 50
        cycle:             1000
        plot only stable: False
        plot decayed:     False
This method adds the following variables to the instance:
a_iso_to_plot      mass number of plotted range of species
isotope_to_plot    corresponding list of isotopes
z_iso_to_plot      corresponding charge numbers
el_iso_to_plot     corresponding element names
abunds             corresponding abundances
isom               isomers and their abundance
Calling get method in cycle mode, adding a_iso_to_plot, z.. el.. isotope.. isoto
pe... to instance
Using the following conditions:
        Atomic mass_range: 1 50
        cycle:             1500
        plot only stable: False
        plot decayed:     False
Finished making abu_chart_0000.png abu_chart_0200.png abu_chart_0400.png abu_cha
rt_0600.png abu_chart_0800.png abu_evolution.png iso_abund_0000.png iso_abund_05
00.png iso_abund_1000.png iso_abund_1500.png plots. Compare with master_results/
*png.
nugrid@ubuntu:~/nugrid/examples/ppn_nova$
```

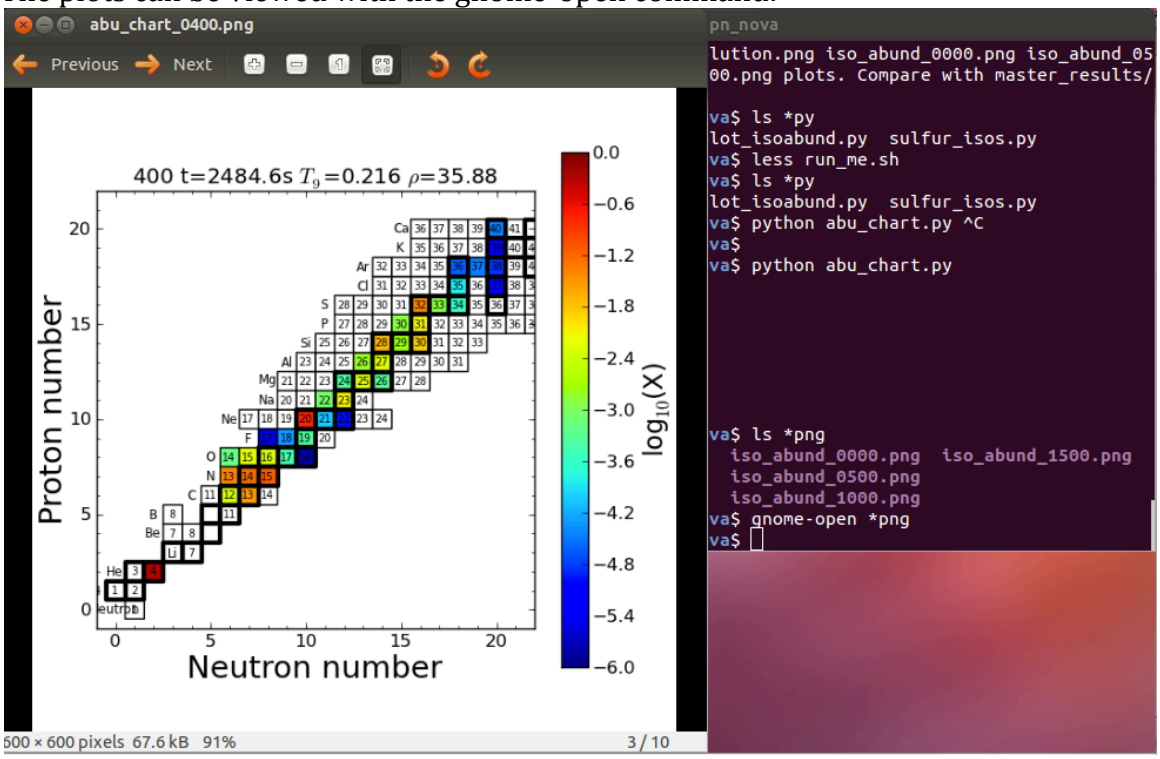
These standard plots are created using the *.py files, which can be executed in batch mode as an argument to python:

```
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ ls *py
abu_chart.py  plot_abu_evolution.py  plot_isoabund.py  sulfur_isos.py
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ python abu_chart.py
```

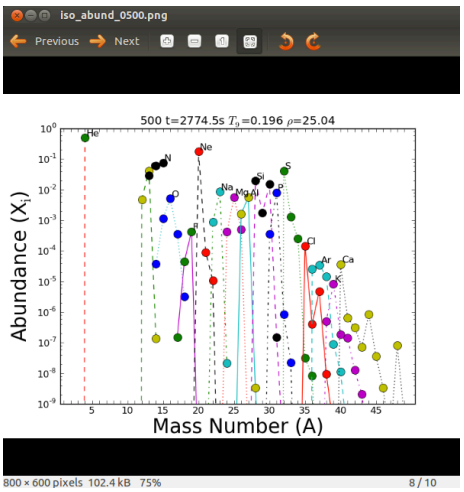
This and the other *py scripts produce the *png image files:

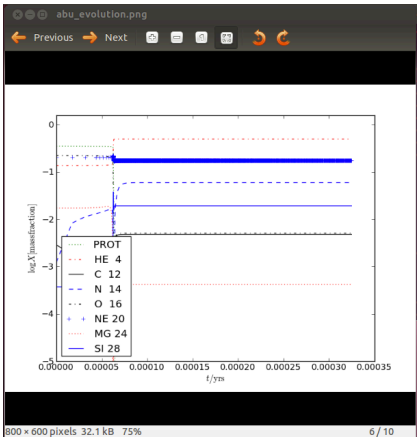
```
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ python abu_chart.py
1911 cycle numbers found in .
Ranging from 0 to 1910
abundance-chart0 is done
abundance-chart200 is done
abundance-chart400 is done
abundance-chart600 is done
abundance-chart800 is done
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ ls *png
abu_chart_0000.png  abu_chart_0600.png  iso_abund_0000.png  iso_abund_1500.png
abu_chart_0200.png  abu_chart_0800.png  iso_abund_0500.png
abu_chart_0400.png  abu_evolution.png   iso_abund_1000.png
nugrid@ubuntu:~/nugrid/examples/ppn_nova$
```

The plots can be viewed with the gnome-open command:



In addition to the abundance chart plots for a number of cycles there are abundance distribution plots as well as abundance evolution plots.





An easy way to work with these plots interactively is to specify the EDITOR environment variable (defaults to 'emacs –nw' which is emacs in terminal mode) and then use the 'ed' method interactively in ipython –pylab (which is aliased as mpython):

```
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ mpython
Enthought Python Distribution -- www.enthought.com

Python 2.7.3 |EPD 7.3-2 (32-bit)| (default, Apr 11 2012, 18:02:54)
Type "copyright", "credits" or "license" for more information.

IPython 0.12.1 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

Welcome to pylab, a matplotlib-based Python environment [backend: WXAgg].
For more information, type 'help(pylab)'.

In [1]: ed plot_isoabund.py
```

```
File Edit Options Buffers Tools Python Help
import ppn
import utils
p=ppn.abu_vector('.')
import data_plot
from matplotlib.pylab import *

mp=p.get('mod')
sparse=500
cycles=mp[::sparse]
form_str='%6.1F'
form_str1='%4.3F'

i=0
for cyc in cycles:
    T9=p.get('t9',fname=cyc)
    Rho=p.get('rho',fname=cyc)
    mod=p.get('mod',fname=cyc)
    time= p.get('agej',fname=cyc)*utils.constants.one_year
    close(i);figure(i);i += 1
    p.iso_abund(cyc,decayed=False,stable=False,show=False)
-UU-:----F1  plot_isoabund.py    Top L1     SVN-3265   (Python)--
```

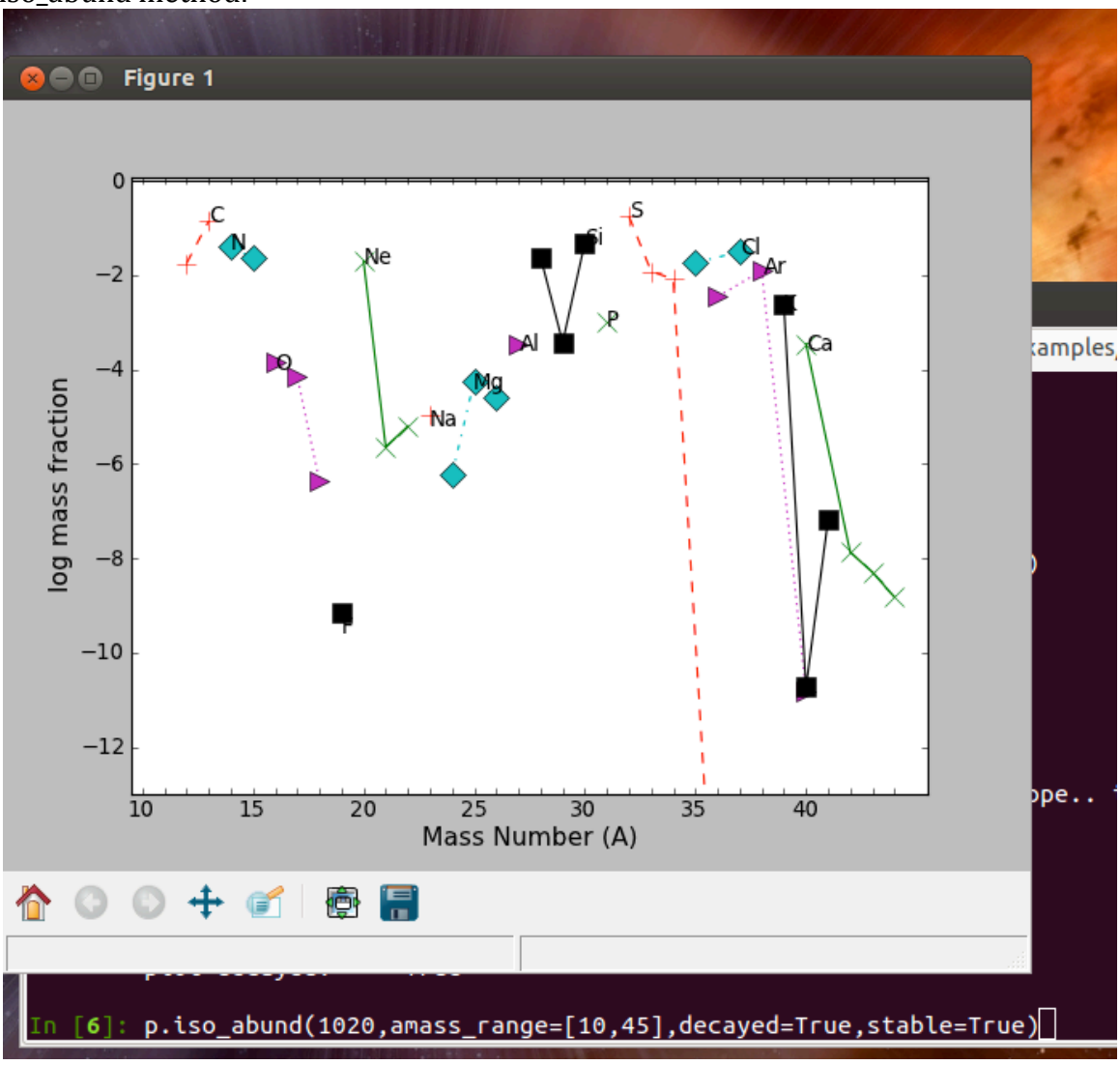
Once saved and exited the script will be executed as if typed in the command line, and all names and variables are available in the current session.

Let's go through the individual steps performed in the script one by one (start an ipython session with ipython –-pylab or the corresponding alias mpython, the ppn module source code can be viewed in $HOME/pylib/ppn.py):

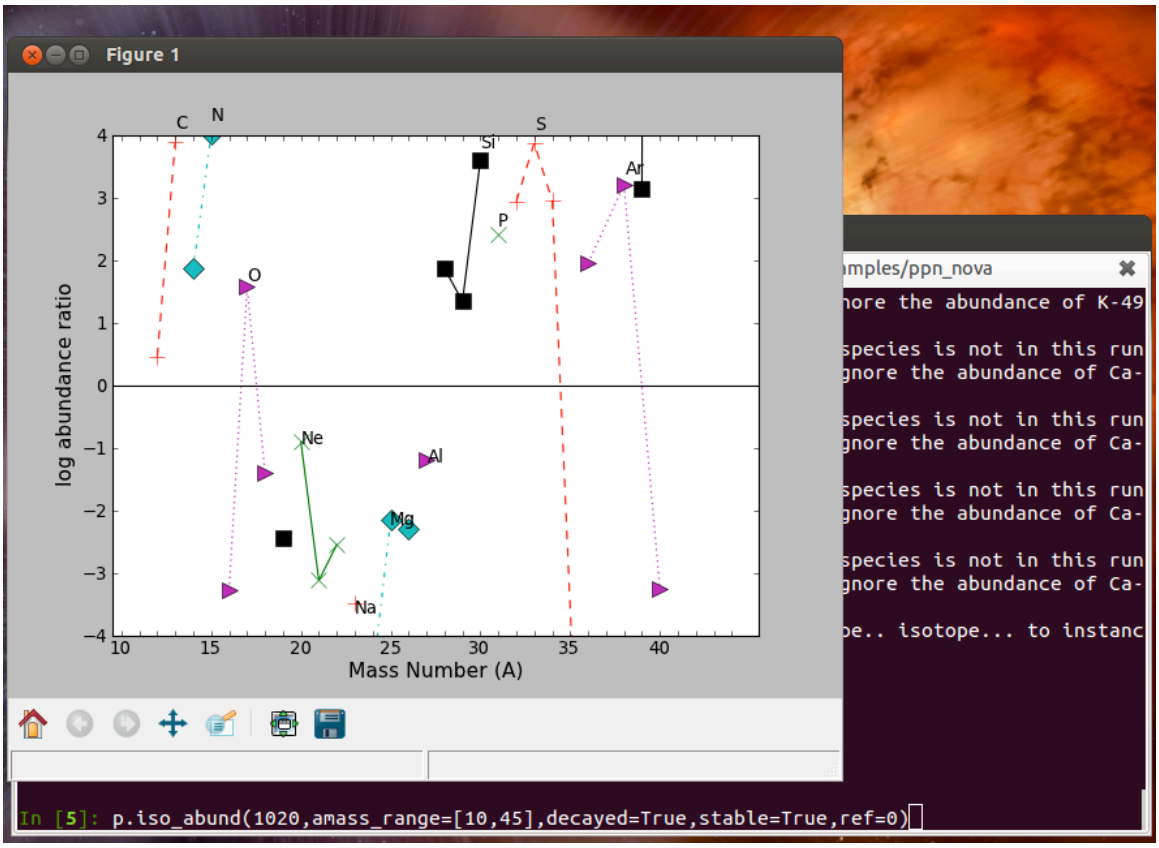
1. import ppn.py module and create an instance of the abundance vector class of data in directory 'standard':

```
In [6]: import ppn

In [7]: p=ppn.abu_vector('.')
1911 cycle numbers found in .
Ranging from 0 to 1910
```

2. In order to perform a plot of an isotopic abundance distribution use the iso_abund method:



3. Often we want to do a ratio plot of the abundance at some cycle to some other cycle, especially maybe the zeroth cycle, which contains the initial abundance. This can be accomplished with the ref=nnn option, where nnn is cycle by which we want to divide:



Next, we would like to perform a reaction rate sensitivity test and plot the result. We would like to know the effect of a reduction of the ^{25}Al(p,γ) by a factor of 100. This can be accomplished by editing the networksetup.txt file. This file is automatically generated during the network setup stage for a new run (ININET=0 in ppn_physics.input). It lists all species and reactions that are actually going to be used according to the selection criteria specified in ppn_physics.input. Instead of creating a new networksetup.txt the code can use a previously created networksetup.txt (ININET=3), which may have been modified. Each reaction rate has a column at the end (last but one) that contains a factor (default 1.0) that will be applied to the rate at run time. The second column contains a boolean (T/F) that allows to toggle the rate. There is also a string that indicates the source of the rate, for this case ILI01 (Illiadis 2001), which may be changed in certain ways.

In order to perform the *effect of a reduction of the ^{25}Al(p,γ) by a factor of 100* test we need to change the factor in the networksetup.txt file for that reaction to 0.01:

```
  540 T  1  AL 25  +  0  00000  ->  1  AL 24  +  1  NEUT    0.100E-98  BASEL  (g,n)   2   1.000E+00  -1.634E+19
  541 T  1  AL 25  +  1  NEUT   ->  1  MG 25  +  1  PROT    0.379E+12  BASEL  (n,p)   3   1.000E+00   4.882E+18
  542 T  1  AL 25  +  1  NEUT   ->  1  NA 22  +  1  HE  4   0.102E+09  BASEL  (n,a)   4   1.000E+00   1.845E+18
  543 T  1  AL 25  +  1  PROT   ->  1  SI 26  +  0  00000  0.146E+01  ILI01  (p,g)   5   1.000E-02   5.324E+18
  544 T  1  AL 25  +  0  00000  ->  1  MG 24  +  1  PROT    0.797E-29  BASEL  (g,p)   6   1.000E+00  -2.192E+18
```

Since the output of the new run will be written into the present directory we have to use the copy_case script first to create a new directory for the present run output and copy all relevant data into that directory.

```
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ ./copy_case.sh standard
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ ed ppn_physics.input
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ ./run_me.sh
 ppn setting up network
 rnetw2008: false species=L   1
```

The one-zone code has three input files, one each for the physics, the solver and the frame:

```
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ ls ppn*input
ppn_frame.input  ppn_physics.input  ppn_solver.input
```

We change ININET to 3 in ppn_physics.input and execute the run_script again.

We use the copy_case.sh script one more time to copy the output to the directory al25pgd100.

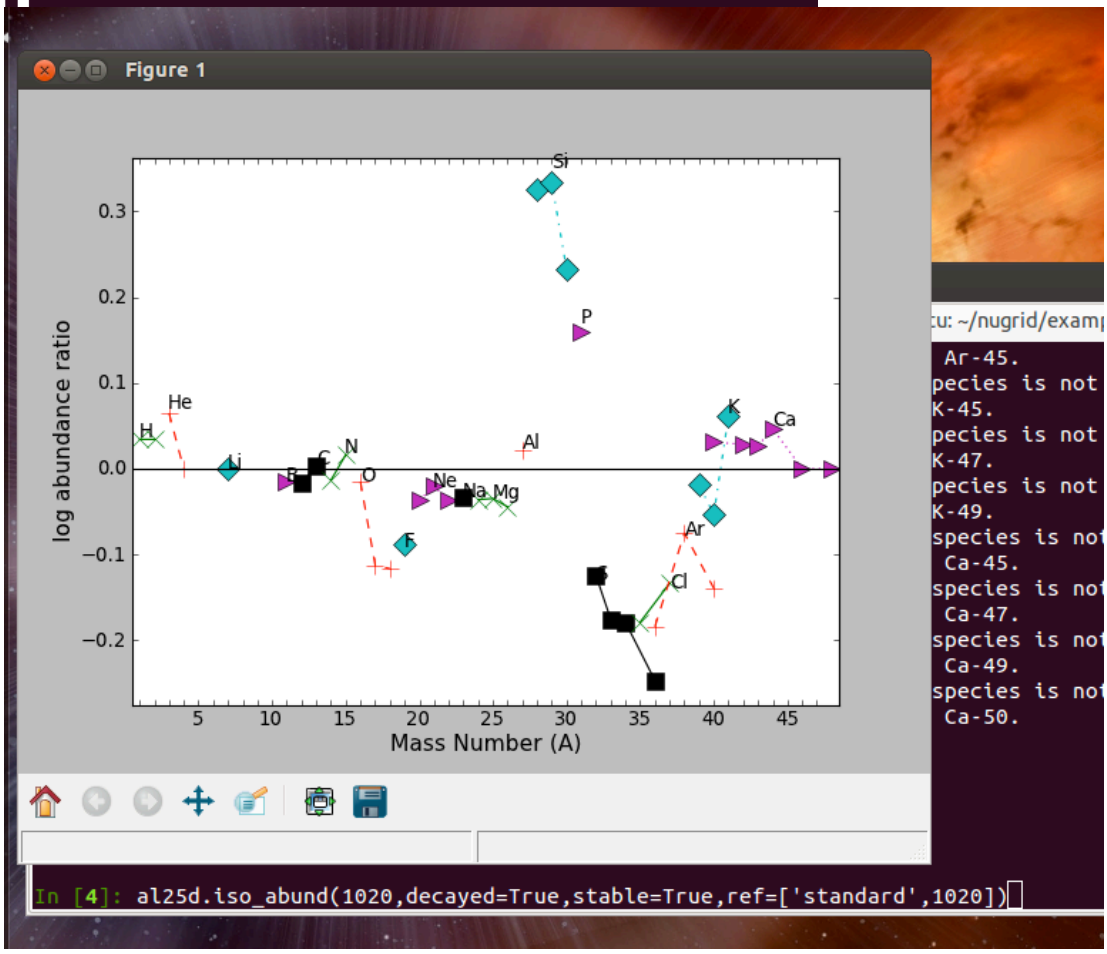
```
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ ./copy_case.sh al25pgd100
```

In order to visualize the change in the abundance distribution that this reaction rate change has we would like to create a ratio plot of the abundance at the last cycle of the new run divided by the abundance of the last cycle of the standard run. First create a ppn.abu_vector instance of the run with the changed reaction rate:

```
nugrid@ubuntu:~/nugrid/examples/ppn_nova$ mpython
Enthought Python Distribution -- www.enthought.com

Python 2.7.3 |EPD 7.3-2 (32-bit)| (default, Apr 11 2012, 18:02:54)
Type "copyright", "credits" or "license" for more information.

IPython 0.12.1 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

Welcome to pylab, a matplotlib-based Python environment [backend: WXAgg].
For more information, type 'help(pylab)'.

In [1]: import ppn

In [2]: al25d=ppn.abu_vector('al25pgd100')
1022 cycle numbers found in al25pgd100
Ranging from 0 to 1021

In [3]: al25d.iso_abund?
```

Then use again the ref keyword, this time in list mode (we may have forgotten how that works so we first check the doc string):

```
Base Class: <type 'instancemethod'>
String Form:<bound method abu_vector.iso_abund of <ppn.abu_vector instan
Namespace:  Interactive
File:       /home/nugrid/pylib/data_plot.py
Definition: al25d.iso_abund(self, cycle, stable=False, amass_range=None,
=True, decayed=False, color_plot=True, grid=False, point_set=1, include_
Docstring:
plot the abundance of all the chemical species

cycle      - a string/integer of the cycle of interest.
             If it is a list of cycles, this method will do a plot
             for each cycle and save them to a file
stable     - a boolean of whether to filter out the unstables.
             Defaults to False
amass_range -a 1x2 array containing the lower and upper Atomic
             mass range. optional. if None plot entire available
             atomic mass range
mass_range - a 1x2 array containing the lower and upper mass range.
             If this is an instance of abu_vector this will
             only plot isotopes that have an atominc mass
             within this range. This will throw an error if
             this range does not make sense ie [45,2]
             if None, it will plot over the entire range
             Defaults to None
ref   - reference cycle. If it is not -1 (default), this method will
```



Now, to get Fig. 15 you need to repeat the same with a run in which you multiply the ^{25}Al(p,γ) rate with 100.